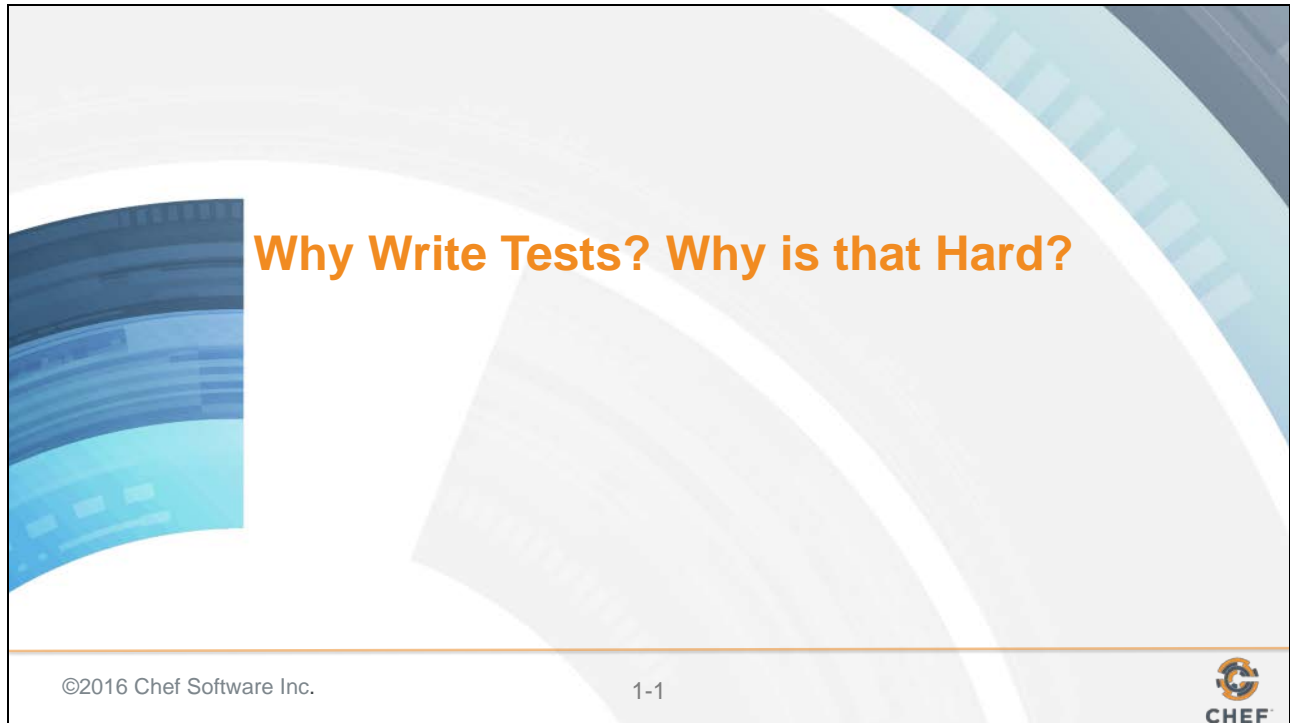**2: Why Write Tests? Why is that Hard?**



Why should you write tests? Why is important that we write tests for the recipes and the cookbooks that we define. Some of you here may be because you are starting to see an importance to what testing can provide. Others of you may not be convinced. Wherever you stand the real reason you came here to learn is to break down the barriers that make testing hard. Because testing is hard!

Slide 2



EXERCISE

## Why Write Tests? Why is that Hard?

*Should I write a test? Perhaps the answer to that question lies in: why write tests?*

**Objective:**

❑ Discussion about Writing Tests
❑ Discussion about Why Writing Tests is Hard

©2016 Chef Software Inc.                    2-2

All of you likely have a personal answer or opinions to these questions. Good. Capture those because we will have a discussion together. To start the discussion I will provide my thoughts and opinions about why I think it is important to write tests. Then I want you to share your thoughts. Then we will discuss the many reasons that testing is hard.

Slide 3



To understand why it is important to write tests I believe it is important to examine the current cookbook development workflow that most individuals employ.
To provide a few answers to why writing tests are powerful and why are they hard to write we need to look at our current cookbook development workflow.

On your local workstation you will write cookbook code. Creating a new recipe to meet new requirements, fixing a bug in an existing recipe, or refactoring complicated recipes into several smaller recipes, helper methods, or maybe even a custom resource.

When you are done with those changes you will spend a few moments visually scanning the code to ensure that your syntax is correct. That every block you start with a 'do' has a matching 'end'. Check your node attributes for spelling issues. Each key-value pair within the hash has a comma that follows.

After enough examination we feel comfortable to upload the cookbook to the Chef Server.

Slide 4



You login to a test node that you patiently bootstrap into a union environment. This is an environment we setup with no cookbook restrictions allowing chef-client to synchronize and apply the latest changes in the recently completed cookbook. Here you see if you got the right package names, spelled all our cookbook attributes correctly, and didn't typo any of the configuration in the templates. If everything converges without error you poke around the system -- running a few commands to see if ports are blocked, services are running, and the logs don't show any errors. Logging out of the working system you feel pretty comfortable promoting the cookbook to the rehearsal environment.

Slide 5



Here in this new environment you may log into another system. Manually perform a chef-client run and then poke around again if everything works. You also may not. It was such a small change and everything worked on the other machine -- so it's likely to work here. Right? Instead of running through a series of ad-hoc verifications again on a new system in this environment - you start to think of the backlog of things that need to get done.

Slide 6



Every time we make changes to our cookbooks we are introducing risk. Ideally we would validate every change to the cookbook but often do not because the amount of time it takes is far too prohibitive. Instead we often will batch up these changes into a set that we will validate. A set of changes like this can often hide errors that we may have introduced. This is definitely true as the complexity of the cookbook code increases.

We have a choice. We can slow down; validating every change. We can also stop making changes altogether. Or we can can adopt new practices, like testing, to help us validate these changes faster; allowing us to continue to move quickly as we continue to satisfy new requirements.

Slide 7
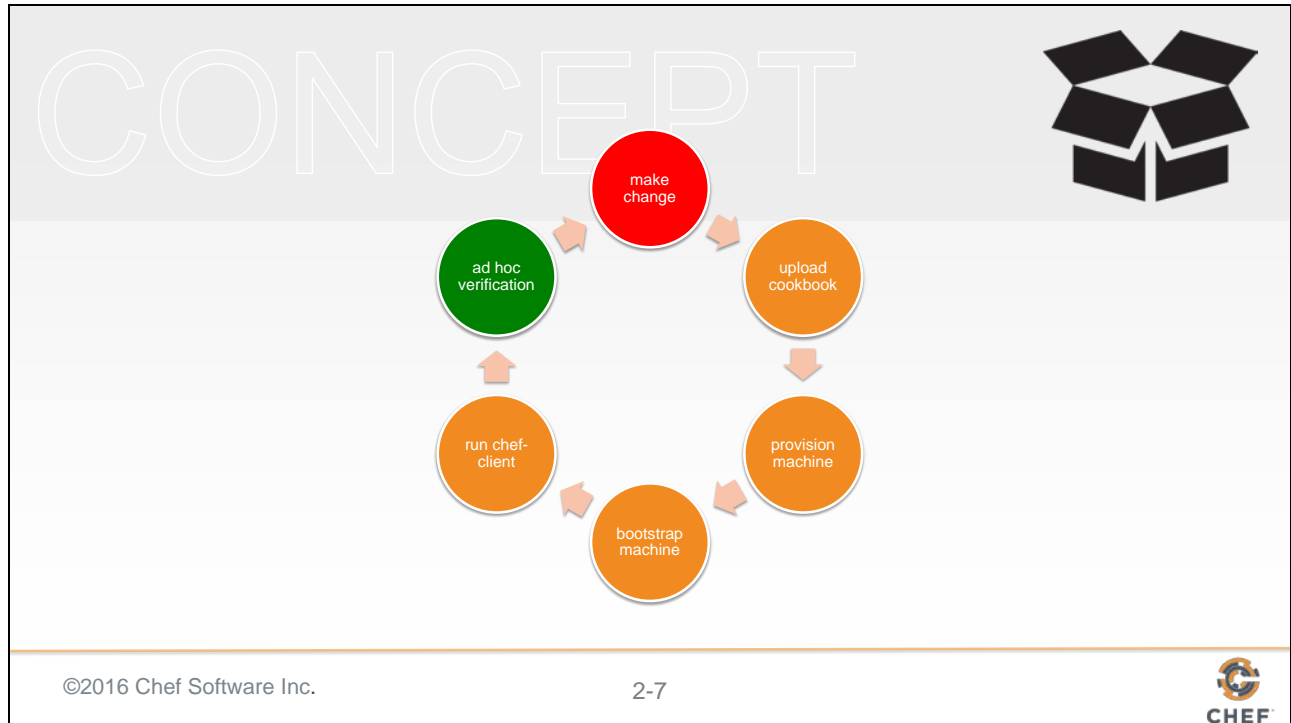


Carrying out testing at every stage (e.g. union, rehearsal) gives great feedback on its success at the cost of the time required for each cookbook to be pushed through this workflow.

Every change needs to be verified in this manner because Ruby, the language Chef is built on, is a dynamically typed programming language. Dynamically typed languages do checking at run-time as opposed to compile-time. This means that ruby files in our cookbook are not executed, thus not validated, until they are run. We also have the problem that we may even write the Ruby correctly but fail to understand the state of the host Operating System (OS) we are attempting to deploy against.

Slide 8
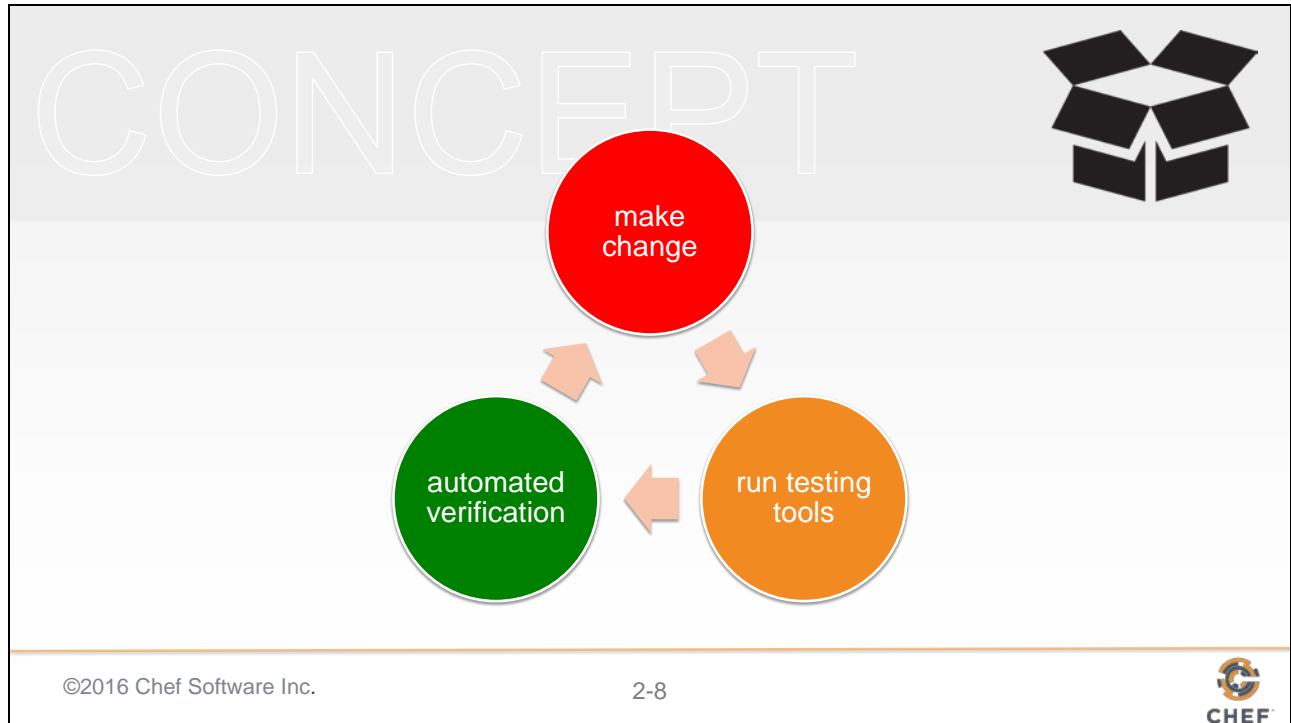


Writing and executing tests decreases the amount of time spent between when you make a change to when you can verify that chance. This reduces the risk within the system.

How testing does address the speed of execution is by removing many of the outside dependencies and allowing you to execute your recipes against in-memory representations of the environment. Or automating the management of virtual machines and the process of executing your recipes against those virtual machines. And second, by allowing you to capture and automate the work that was previously performed in ad hoc verification.

Slide 9



I shared with you my opinion on why I think it is important to write tests. Now I would like to understand what reasons you see for writing tests. I would also like to know your reasons for not writing tests.

Instructor Note: With large groups I often find it better to have individuals turn to the individuals around them, form groups of whatever size they feel comfortable, and have them take turns asking and answering the questions. When all the groups are done I then open the discussion up to the entire group allowing each group or individuals to share their answers.

Slide 10



EXERCISE

## Why Write Tests? Why is that Hard?

*To test or not to tests. It's no longer the question. Now I need to think: what makes writing tests challenging?*

**Objective:**

✓ Discussion about Writing Tests
❏ Discussion about Why Writing Tests is Hard

You may or may not be convinced that there is value in writing and executing tests. If the opinions we all have expressed has not convinced you I encourage you to continue to find more discussions where you can hear more opinions and share yours with others. It is important to have these discussions within your teams and your organization.

I want to now focus the discussion on the reasons why writing tests are hard. Similar to the previous discussion I want to provide my opinion to start the discussion. I want you to also contribute your opinions and experiences as they are equally valuable.

Slide 11



The language you use to define your tests in is not the same as the language you use to compose your original intentions. To test your code you need to write more code. However, this new code that you write is different as you are expressing your desired expectations of the system across a number of scenarios. This requires you to learn one or more new languages which have completely new systems and structures.

Testing asks you to solve a different problem in a different order when compared to process of writing software. You have to overcome particular challenges created by an implementation and express the desired expectations of that implementation before it is even built.

Slide 12

CONCEPT

## Another Workflow

Executing tests requires learning new tools, commands, flags and configurations with entirely new mechanisms that provide you feedback.

©2016 Chef Software Inc.          2-12

Testing also asks you to change your behaviors through the new tools required to execute the tests. These tools represent a huge domain of knowledge expressed in all the commands, flags, and configuration that must be understood to be used correctly and then effectively as the complexity of your testing tools grow. The largest, and most immediate impact is on your development workflow which has to adopt new steps that feel unsure and even more unreliable as you receive a barrage of feedback in unfamiliar formats.

Slide 13



I shared with you my opinion on why I think it is hard to write tests. Now I would like to understand what reasons you see that make testing hard.

After we have expressed a set of reasons we should leave time within the discussion to discuss ways in which you have made it less hard.

Instructor Note: With large groups I often find it better to have individuals turn to the individuals around them, form groups of whatever size they feel comfortable, and have them take turns asking and answering the questions. When all the groups are done I then open the discussion up to the entire group allowing each group or individuals to share their answers.

Slide 14



With our two discussions complete lets pause now for any questions that were not covered or even came out of the discussions.

Slide 15



Before we complete this section and start learning some of these new tools and languages let us pause for questions.

Slide 16